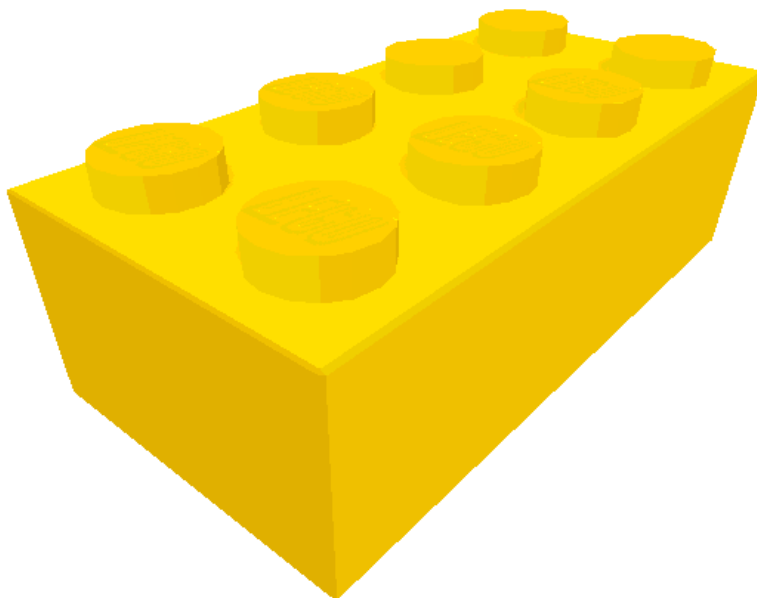


LEGO CREATOR:

How to create a brick.



Contents

1. INTRODUCTION	2
2. OBJECT HIERARCHY & NAMING	2
3. DISTANCING.....	2
4. SHAPE USAGE	3
5. HOW TO BUILD A LEGO BRICK	4
6. TIPS AND TACTICS	5
APPENDIX A: HOW TO BUILD A LEGO BRICK CHECKLIST	7
APPENDIX B: BRICK DIMENSIONS (AS USED IN THE SHAPE EDITOR)	8
APPENDIX C: TYPICAL TEXTURE CO-ORDINATE VALUES	9
APPENDIX D: PALETTE COLOURS AND CODES.....	10
APPENDIX E: LEGO DECALS LIST	11
APPENDIX F: EXAMPLE OF STUD INFORMATION	13
APPENDIX G: EXAMPLE OF COLLISION INFORMATION	14
APPENDIX H: GLOSSARY	15

1. Introduction

This document describes the method for creating bricks that will be used in Lego Creator. Presently the method is to create the brick and all it's relevant information in VRT, save the brick as a .vca and then transform it as a .lca.

There are few files needed for the various steps of creation and test.

The template from which you start building the brick is currently **ClipArt4.svr**. Once the .vca file is saved, the brick has to be tested in **studtestVCA.svr**. To convert the .vca to .lca, use **vca2lca.svr**.

2. Object Hierarchy & Naming

A brick should consist of a group containing one or more child objects. These child objects should not be placed in sub-groups of the brick. The brick group should be named as the brick code specified in the LEGO catalogue prefixed by an 'L' (e.g.) 'L335800'. Textured bricks are named as blank bricks followed by a texture reference number. The name of each child object should be the brick group name suffixed by 'PART 1', 'PART 2', 'PART 3', etc. depending upon the number of child objects.

A shape used to force automatic z-buffering should be suffixed with the letter 'Z'. Similarly, a polyhedra collision shape should be suffixed with the letter 'C'.

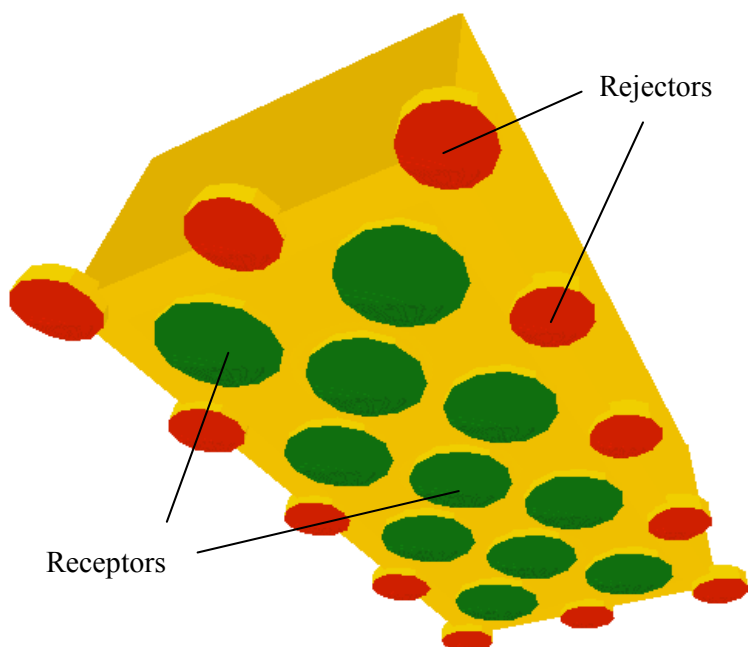
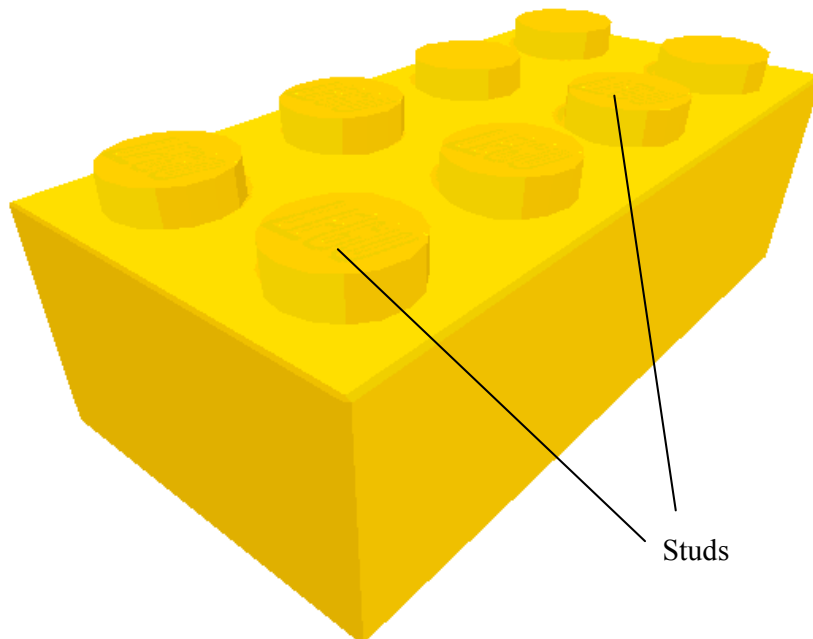
Where a brick is actually constructed by placing two or more individual LEGO bricks together, the brick group should be given an upper-case name describing the brick, prefixed by a 'C' (e.g.) 'C FLAG', 'C MOTORCYCLE CHASSIS'. The naming of child objects should follow the convention described above (e.g.) 'C FLAG PART 1'.

3. Distancing

Each child object of a brick may be distanced if required. The name of the distanced object should be the name of the object that it replaces with a 'D1' appended at the end. Where objects are used to represent further levels of distancing, their names should be suffixed by 'D2', 'D3', etc. to indicate the level of distancing.

4. Shape usage

A separate shape must be used for each non-distanced object that has a unique arrangement of studs, receptors and rejectors.



The same shape can be used for any number of replacement objects. Replacement shapes do not carry any stud, receptor or rejector information. Shapes should be named by prefixing the brick code in the LEGO catalogue with 'L', as for objects, but should be suffixed by a capitalised description of the shape (e.g.) 'L 338650 Upper Frame'.

5. How to build a LEGO brick

Open the ClipArt4.svr file.

First, start building the close-up shape geometry, making sure that all facets sort correctly. All the brick dimensions are indicated in Appendix B. Facet switching may be required in some cases. The bevels are used whenever it will improve the look of the brick when joined to another brick. If studs are required on a surface which lies within the bounding cube, it may be required to use inside facets to enable the studs to be viewed correctly (the studs are child objects of the shape). While modelling the underneath of the brick, keep in mind that a texture of the receptor area will be applied. The receptor area is surrounded by facets on the same plane (these are called receptor edge facets).

The next step is to model the distanced object. The number of distanced models of an object has to be chosen bearing in mind the limits of the draw-list.

The distanced object doesn't have any bevels or receptor texture.
The usual distancing value is 1000 units.

Group all shapes as direct children of the brick group.

Collision cuboids have to be applied to the brick group, if the object doesn't have facets at an angle. Objects with facets at an angle require special collision information (e.g. roof-tiles) (the method will be described later on). Both methods cannot be mixed.

Resize the object to a tenth of its original size by deleting the last number in the size boxes under Position => Standard Attributes. Do not use the Rescale function, as it may come up with some imprecise change of size, which may go unnoticed.

Apply the textures to the object, and manipulate them to fit the object. Typical values are given in Appendix C.

Cut out the textures used attribute, they will be applied via the world script (Objects => Attributes => Cut...).

Edit the world script to correct the texture reference on the object. The texture numbers are given in Appendix D and E.

Flag all objects as enterable except the brick group.

Add rotation attributes to all objects except the brick group.

For each object, create four animation cells, which will correspond to the various orientation colourings of the brick.

Apply animated colours (in the shape editor).

Press 'R' for launching the auto lighting of the four cells.

Change the animated colours (from no.8 to no.5) on the receptor edge facets (facets which surround the receptor texture), in the facet editor.

In the world editor, apply 'Current colours' by sampling a facet and then colouring it the same colour.

Remove animated colours four times in the world editor (Appearance => Animated Colours...=> Delete).

Apply Stud information in Shape editor (not to the distanced objects). An example is given in Appendix F.

If required, add Collision information in the Shape editor (to the distanced shape or to the collision shape). A collision shape has to be created if the brick is concave, if not the distanced shape will do. The collision shape will be made of a collection of convex blocks, each of different colour. The collision object will be flagged as invisible in the world. An example is given in Appendix G.

Save the object as a .vca.

Import the .vca in studtestVCA.svr and test the brick (stud information, textures, various colours, etc.).

When all is correct, transform the .vca into a .lca, using vca2lca.svr. Make sure to check the properties on the cube. Update the text file referenced in the properties (the number corresponds to number of .vcas to transform, followed by their names). Run the world and click on the cube.

6. Tips and tactics

6.1 General

In Creator, the bricks move on a 40 by 40 grid horizontally, and 32 vertically. The group holding the shapes for a brick must conform to this grid, and the shapes inside it that contain linking information must also be 'on grid' within the group.

All bricks are to be built so that the LEGO embossing on the studs is in the same orientation as all the other bricks. In world editor plan view in VRT, this translates as reading from left to right.

Some LEGO bricks have studs that are drilled to allow things to be stuck into them. These are treated as normal studs for LEGO Creator.

Some of the more basic bricks are built with a small fillet to enhance their appearance. As this can involve facets which are very long compared to their width, it sometimes causes problems

with flickering facets and should therefore be used with care, and only where it would make a definite visual improvement.

6.2 Avoid meshing

The reason's for avoiding meshing is a bit involved, but here goes;

Reason 1.

Meshing does (more or less) the same job as Z-buffering.

We have automatic Z-buffering turned on in Creator, so that when bounding cubes intersect, the bricks Z-buffer.

In most cases, as soon as a brick is connected to another, an intersection occurs, and the shapes are Z-buffered.

In a meshed shape where the studs appear within the bounding cube, they are drawn over by the brick unless inside facets are used (as children of the brick). But other studs then show through facets which have been made 'inside facets' for this reason.

Reason 2.

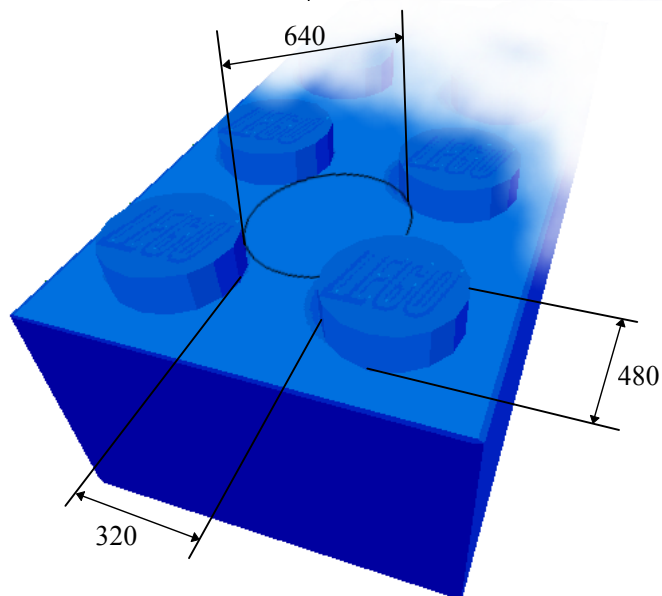
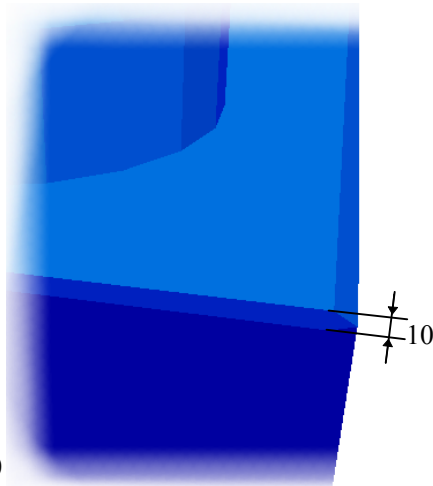
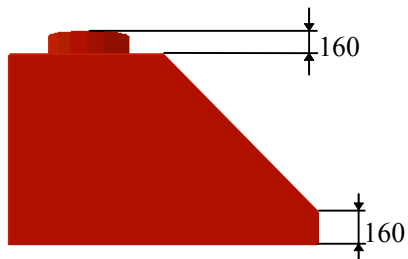
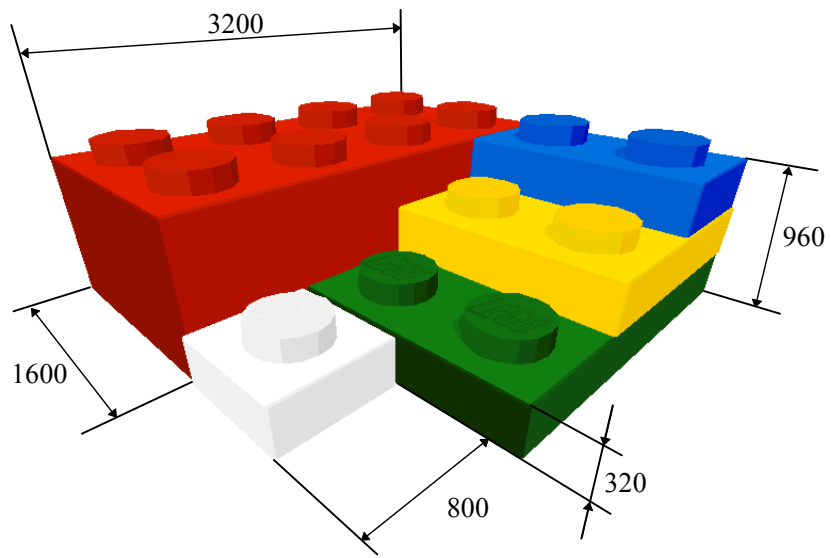
Z-buffering is accelerated by 3D graphics cards, meshing is not (at least, that's how I understand it).

Therefore, I tend to use a Z-buffer object (a shape which is totally inside another, but is a sibling of it) to force Z-buffering to occur all the time to a brick which would Z-buffer anyway as soon as it was attached to another brick.

Appendix A: How to build a LEGO brick checklist

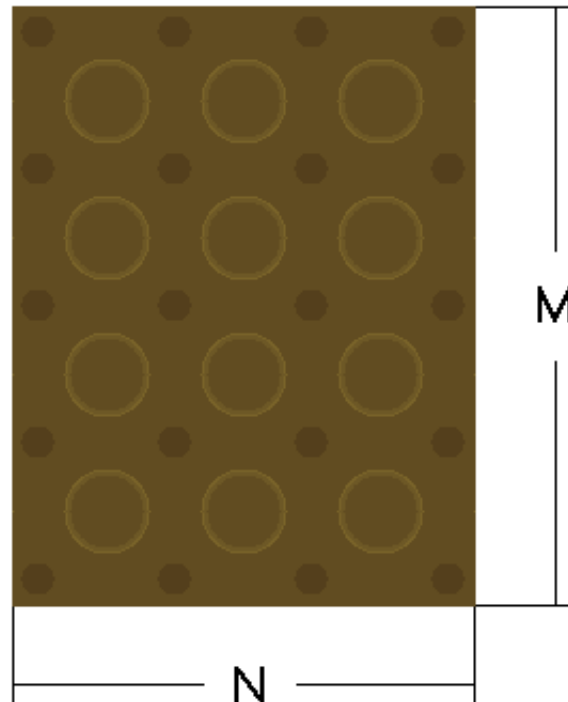
1. Build the geometry.
Concavities with studs may require inside facets.
Collision cuboids to be on holding group.
2. Build distanced objects and apply (typically 1000 for first replacement).
3. Resize. 1/10 by deleting the last zero in the Standard Attributes dialog box (Position => Standard Attributes...).
4. Add stud and receptor textures.
5. Cut 'Textures Used Attribute' (Objects => Attributes => Cut...).
6. Edit world script to correct texture references.
7. Flag objects as enterable - Not holding group.
8. Add rotations to objects - Not holding group.
9. For each object, make four animation cells.
Apply shape animated colours (select animated colours in shape editor).
Press 'R' for auto lighting of the four cells.
Change receptor edge facet (underside) colours (from no.8 to no.5).
 Find facet number from facet editor.
 Select 'Shape - Animated colours'.
 Find correct facets and re-colour (use 'Alt-n' to advance cell number).
10. Apply 'Current colours' by sampling a facet and then colouring it the same colour.
11. Remove animated colours in World editor, do not cut. (Appearance => Animated colours => delete (four times)).
12. Apply Stud information in Shape editor (not to distanced objects).
13. Add Collision information in the Shape editor (to the distanced object or to the collision shape).

Appendix B: Brick dimensions (as used in the shape editor)



Appendix C: Typical texture co-ordinate values

Type 1



The array below corresponds to a typical set of co-ordinates for a type 1 texture, where $M=4$ and $N=3$ (as shown in the above figure).

0.82	0.82
0.82	N.18
M.18	N.18
M.18	0.82

Type 2



The array below corresponds to a typical set of co-ordinates for a type 2 texture, where $L=4$ (as shown in the above figure).

L.18	0.18
0.82	0.18
0.82	0.82
L.18	0.82

Appendix D: Palette colours and codes

		Texture Reference					
		Studs				Receptors	
		0	90	180	270	Type 1	Type 2
Yellow	YE	1	61	81	101	21	41
Red	RE	2	62	82	102	22	42
Blue	BL	3	63	83	103	23	43
Green	GR	4	64	84	104	24	44
White	WH	5	65	85	105	25	45
Black	BK	6	66	86	106	26	46
Light grey	GY	7	67	87	107	27	47
Dark grey	GD	8	68	88	108	28	48
Mint	MI	9	69	89	109	29	49
Pink	PK	10	70	90	110	30	50
Rose pink	PR	11	71	91	111	31	51
Brick yellow	YB	12	72	92	112	32	52
Light yellow	YL	13	73	93	113	33	53
Light mauve	LM	14	74	94	114	34	54
Brown	BR	15	75	95	115	35	55
Transparent white	TW						
Transparent yellow	TY						
Transparent red	TR						
Transparent green	TG						
Transparent blue	TB						
Transparent light blue	TL						

Appendix E: LEGO decals list

141	Tyre tread	181	
142	Yellow wheel centre	182	Backlight 1 Red
143	White wheel centre	183	
144	Exhaust stack ribs	184	Backlight 1 White
145	Sign - Crossing	185	Backlight 1 Black
146	Sign - Stop	186	Police - Black
147	Sign - 40	187	Police - White
148	Stripe Red and white	188	Police Shield Black
149	Stripe Red and blue	189	Police Shield Grey
150	Sign - No left turn	190	Telecom BW
151	Band Red on blue	191	Telecom COL
152	Stripe Green and white	192	Wanted 1
153	Stripe Black and yellow	193	Wanted 2
154	100 Dollar	194	Phone 1
155	4WD	195	Phone 2
156	4WD Bison	196	Reel 2 reel
157	Bison	197	Post box
158	Burger	198	Recycle
159	Café	199	Pizza
160	Letter	200	Cash machine
161	Grill 1 Yellow	201	Octan 1
162	Grill 1 Red	202	Qwerty 1
163	Grill 1 Blue	203	Qwerty2
164	Grill 1 White	204	Cargo 1
165	Grill 1 Black	205	Cargo 2
166	Grill 2 Yellow	206	
167		207	
168	Grill 2 Blue	208	Turbo 1
169	Grill 2 White	209	Turbo 2
170		210	Turbo 3
171		211	Fire shield Red
172		212	Console 1
173		213	Console 2 (Fire engine)
174		214	Grey wheel centre
175	Grill 3 Black	215	Console 4
176	Grill 4 Yellow	216	Console 3
177		217	Bike 1
178	Grill 4 Blue	218	Bike 2
179		219	Hot car
180		220	Monitor

221	Podium 1	261	Cargo Tail Port
222	Podium 2	262	Cargo Tail Stbd
223	Podium 3	263	Octan flag
224	Racing 73 - 1	264	Bush branches
225	Truck side 1 L	265	Camera (Button)
226	Truck side 1 R	266	Minifig (Button)
227	Truck side 2 L	267	Number 1
228	Truck side 2 R	268	Number 2
229	Racing 71 - 1	269	Console 7
230	Racing 71 - 2	270	Console 8
231	Number 5	271	Stripe Black & Grey
232	Sign - Keep left	272	Cone (road)
233	Console 5	273	Blank
234	Console 6	274	Target
235	Shield Dollar Black	275	No Target
236	Shield Dollar Grey	276	Lego logo for AVI
237	Window Dollar	277	Pencil
238	MiniFig - Bodies 1	278	Console 9
239	MiniFig - Faces 1	279	Helmet - Star
240		280	Helmet - Flame Green
241		281	Helmet - Flame Red
242		282	Helmet - Flame Blue
243		283	Biker Bob's Jacket
244	Ground - Police	284	Biker Bob's Face
245	Ground - Truck	285	
246	Grey panel	286	
247	Police Shield White	287	
248	Grid - GD	288	
249	Grid - GY	289	
250	Wood crate side	290	
251	Check Black and White	291	
252	Fire shield White	292	
253	Window - Police	293	
254	Window - Burger	294	
255	Window - Fire	295	
256	Window - Octan	296	
257		297	
258		298	
259	Dynamite	299	
260	Truck side 3	300	

Appendix F: Example of stud information

(the text is attached to the close-up shape. Shape => Edit text... In this case the brick is a 4 by 2 brick.)



TOP: 3	Top surface is at a height of 3 slim bricks.
Type: +STUD	Add studs.
Pos: 0 0	Position of the stud area (in half studs).
Size: 5 9	Size of the area is in half studs plus one (due to rotations).
BOT: 0	Bottom surface is at a height of 0 slim bricks.
Type: +SINGLE	Add single type receptors.
Pos: 0 0	Position of the receptor area (in half studs)..
Size: 5 9	Size of the area is in half receptors plus one (due to rotations).
TUBE:0	Tubes' surface is at a height of 0 slim bricks.
Type: +SINGLE	Add single type tubes.
Pos: 1 1	Position of the tube area.
Size: 3 5 7	Size of the area is in half tubes plus one (due to rotations). The second value is the tube height in stud heights.
COL: { YE RE BL GR WH BK GY GD MI PK PR YB YL LM BR TW TY TR TG TB TL }	Colours available for that brick. In this case: yellow, red, blue, green, white, black, light grey, dark grey, mint, pink, rose pink, brick yellow, light yellow, light mauve, brown, transparent white, transparent yellow, transparent red, transparent green, transparent blue, transparent light blue. Only facets that are initially the first colour in the list will be recoloured.

Appendix G: Example of collision information

(This text is attached to the collision shape, which can be the distanced object. Shape => Edit text... In this case, all the facets will be yellow.)

VOL: { 1 YE }

The '1' is the number of the animation cell used for reference. When the distance object is also used as a collision object, this value is typically '5'.

The colour references that follow are the colours of the separate parts of the collision shape that form enclosed convex volumes.

Appendix H: Glossary

Anti-stud: The anti-stud is a cylinder, which can fit in a stud-gap or on a stud.

Facet Switching: This is a method used to make some awkward shapes sort correctly (a cross shape is a good example). Two identical facets appear in different positions in the facet editor. The second would always overlay the first except that it is associated with a facet in a different position/orientation which causes the second facet to vanish (switch) and the first facet show instead when viewed from certain angles.

Receptor: The receptor is the spacing underneath the brick, which accepts a stud.

Rejector: The rejector is the surface underneath the brick, which doesn't accept a stud.

Stud: The stud is the round extrusion on a brick's surface, which allows you to attach another brick above it.

Stud-gap: The stud-gap is the spacing between four studs, which allows the positioning of an anti-stud

Z-Buffer object: A shape which is totally inside another, but is a sibling of it, to force Z-buffering to occur all the time to a brick which would Z-buffer anyway as soon as it was attached to another brick.